EPiC Series in Engineering
Volume 3, 2018, Pages 1703–1712
HIC 2018. 13th International Conference on Hydroinformatics

Shape optimization of hydraulic structures: an example of an optimum design of a fish passage

P. Prodanovic[1*], C. Goeury[2], F. Zaoui[2], R. Ata[2], J. Fontaine[2], P. Tassi[2] and Y. Audouin[2]
[1]Saint-Venant Laboratory for Hydraulics, Chatou, 78401, France,
[*]now with Riggs Engineering Ltd., London, Ontario, N6K 1C7, Canada
[2]EDF R&D National Laboratory for Hydraulics and Environment, Chatou, 78401, France
pprodanovic@riggsengineering.com, cedric.goeury@edf.fr, fabrice.zaoui@edf.fr, riadh.ata@edf.fr, jacques-j.fontaine@edf.fr, pablo.tassi@edf.fr, yoann.audouin@edf.fr

Abstract

This paper presents a practical methodology developed for shape optimization studies of hydraulic structures using environmental numerical modelling codes. The methodology starts by defining the optimization problem and identifying relevant problem constraints. Design variables in shape optimization studies are configuration of structures (such as length or spacing of groins, orientation and layout of breakwaters, etc.) whose optimal orientation is not known a priori. The optimization problem is solved numerically by coupling an optimization algorithm to a numerical model. The coupled system is able to define, test and evaluate a multitude of new shapes, which are internally generated and then simulated using a numerical model. The developed methodology is tested using an example of an optimum design of a fish passage, where the design variables are the length and the position of slots. In this paper an objective function is defined where a target is specified and the numerical optimizer is asked to retrieve the target solution. Such a definition of the objective function is used to validate the developed tool chain. This work uses the numerical model TELEMAC-2Dfrom the TELEMAC-MASCARET suite of numerical solvers for the solution of shallow water equations, coupled with various numerical optimization algorithms available in the literature.

1 Introduction

Applications of shape optimization in hydraulic engineering are rare, especially when those involving cases where optimization was used to inform actual engineering design of real life civil works. There are examples found in the literature (citations provided in Section 2) where shape

optimization problems are solved, but using the simplest of numerical codes. Use of simple models is generally not sufficient in completing detailed designs of costly civil works, or evaluating impact of hydraulic structures in environmentally sensitive areas. The main intent of this work is to present a methodology that can be applied by end users working on shape optimization problems in the domains of river and coastal engineering. For example, applications of shape optimization could include determining the optimal layout of a groin field along a coastline that could address sedimentation and navigation issues, the shape and orientation of a breakwater protecting a harbour or a marina subject to various environmental and economic constraints, as well as many others.

A shape optimization study in hydraulic engineering (civil and coastal engineering) requires the end user to couple a numerical model that captures sufficiently enough physics with an optimization algorithm, where the optimization process generates different shapes of the structures, simulates the system (solves the governing equations), and iteratively adjust the shape while respecting predefined constraints. Carrying out such a simulation requires an automatic way to generate a new shape (which, in terms of numerical modelling means generating a new model geometry) at each iteration of the optimization. The numerical optimizer then carries out the iterations until a set criteria is reached (such as percent change in the objective function value, prescribed number of iterations, etc.). It is the understanding of the authors that such tools are not widely available or presently used in design of major civil works in the realm of river and coastal engineering.

1.1 Literature review

In the text that follows, select examples of shape optimization found in the literature are summarized.

Elchahal et al. (2013) present a methodology developed to optimize the layout of the detached breakwaters in a port. The paper presents the methodology where evolutionary algorithms are coupled to a simple phase resolving numerical model (solution of the mild-slope equation) to determine an optimal layout of breakwaters in a port subject to wave agitation and navigation constraints. The study by Erpicum et al. (2009) provides an example of an application of shape optimization in the design of a guide wall at a channel that feeds a hydropower plant. In the application case provided the orientation of the intake channel at the reservoir creates an undesirable zone of re-circulation that capture debris floating on the water surface. Alvarez-Vasquez et al. (2009) present an example of shape optimization of the design of a fish flume, a hydraulic structure placed at dams and weirs to assist the upstream migration of fish. The problem considered is a design of a vertical slot fish flume (i.e., a rectangular channel divided into a number of compartments or slots). The objective function is defined as a square of the difference between the velocity of the shape evaluated by the optimizer and the desired velocity specified by the user. The problem's constraints were set by specifying the minimum distance between the vertical slots. A method based on the Nedler and Mead's (1965) simplex algorithm is used to carry out the optimization procedure.

A common thread among the above cited studies is that each uses a particular optimizer to produce an optimal shape. To the best knowledge of the authors, a study has not yet been reported in the literature where the same shape optimization problem is solved using different numerical optimizers.

2 Methodology

2.1 General sketch of the shape optimization process

This subsection provides a basic sketch of the shape optimization process that is implemented in this study. The end goal of this work is to provide end users with tools to carry out shape optimization problems on their own applications. The general steps of shape optimization studies are:

1. Create a numerical model of a general shape of the problem under study,
2. Devise an objective function for the shape optimization problem, and identify appropriate problem constraints,
3. Launch the optimization process:
    a. Choose an initial state of the optimization problem (i.e., start with some initial shape, or allow an optimizer to select a starting shape),
    b. Simulate the numerical model and obtain a valid solution corresponding to the chosen shape,
    c. Extract model results for the shape simulated, and evaluate the objective function value (and/or constraints),
    d. Allow the optimization algorithm to select a new shape of the problem, and
    e. Repeat steps 3b) – 3d) until an optimum is found.

The above procedure requires automation of tasks that are typically not available with off-the-shelf commercial or even open source numerical models. The methodology requires coupling a numerical simulation model with an optimization algorithm, which then jointly are able to create and evaluate the fitness of each generated shape according to the specified criteria. To be able to carry out such simulations, a tool set is required that can automatically generate new shapes (which requires creating a new mesh, assigning bathymetry/topography to the mesh, assigning initial and boundary conditions) that are ready to be used in a numerical model. A scripting set of tools are then needed to take the generated shape and obtain a valid result using a numerical model. An automated way of extracting the numerical model results are used to evaluate the objective function, which are given to the optimizer so that it can select a new shape. The process is repeated until an optimum is found.

2.2 Numerical modeling

The TELEMAC-MASCARET numerical modelling system is used in this work.Most of the modules in the TELEMAC-MASCARET system use the theory of finite elements to discretize the spatial domain of the problem (some modules also give the user access to finite volume schemes as well). Triangular elements are used to define the computational domain on which the governing equations are solved. Each module in the system provides the end user with various methods that can be used to solve the governing equations in the computational problem under consideration (flow and momentum equations for hydrodynamics of free surface flows, spectral action balance for wave propagation, sediment balance equations for sediment transport, etc.).

2.3 Pre- and post-processing tool chain

The pre- and post-processing tool chain developed in this work uses the PPUTILS toolkit (Prodanovic, 20117). PPUTILS is an open source library of code developed in the Python

programming language that provides users with command line tools to carry out basic numerical modelling tasks such as mesh generation, assigning properties to the mesh (bathymetry and friction zones), generation of initial and boundary conditions, extraction of data from result files and visualization of model results. Each script in the PPUTILS toolkit is written to efficiently solve only a particular task. Chaining multiple such scripts allows the user to carry out the required modelling tasks using select scripts from the PPUTILS library.

2.4 Numerical optimization

Rather than relying on a single numerical optimizer this work has been structured to allow the user to select among a number of different numerical optimizers. In this work a number of different libraries of numerical optimizers are used, including: Scipy (2017), Mystic (2017), and Pyswarm (2017). Mystic optimizers are further described in McKerns et al. (2011). More specifically, the following numerical optimizers were used:

1. Nedler and Mead simplex solver (fmin),
2. Powell's (modified) level set method (fminpowell),
3. Uniform random distribution of N solvers (buckshot),
4. Distribution of N solvers on a regular grid (lattice),
5. Price and Storn's differential evolution algorithm (diffev),
6. Sequential Least SQuare Programming (SLSQP) optimizer,
7. Particle swarm optimizer (PSO),
8. Latin Hypercube Sampling (LHS), also referred to as the brute force method.

Optimizers 1-5 are available from Mystic (2017), optimizer 6 from Scipy (2017), while optimizer 7 is available from the Pyswarm (2017) library. The above optimizers are all available as open source code.

Note that LHS is not an optimizer, but rather a statistical method to generate a large number of parameter values from a multi-dimensional space. In our case, the LHS technique was used to produce a large set of shapes of the fish flume, which were then filtered to remove shapes that did not meet the problem constraints.

3 Application

3.1 Optimal design of a fish passage

The methodology presented in above is illustrated by a test case of a design of a fish passage (also referred to as a fish ladder or a fish flume). Fish passage structures are typically designed to enable fish to pass through a barrier (such as a dam or a weir) by swimming and leaping up on a designed sloping channel divided into a number of compartments. The velocity of water in the pools has to be high enough to attract the migrating fish, but cannot be too great to wash the fish downstream or otherwise harm it or prevent it from using the structure.

The shape optimization problem consists of finding an optimum position of the slots of a typical fish passage. The geometry of the fish passage used in this example is obtained from Alvarez-Vasquez (2009), where a similar problem is solved using a numerical flow solver linked to a single optimizer. The fish passage consists of ten identical compartments, with each having two slots. The slots are referred to in this work as upper and lower. The optimization problem is to select the position and length of the upper and lower slots, such that it optimizes the objective function while satisfying the problem constraints.

A sketch of the fish flume is shown in Figure 1 where relevant dimensions are shown. Other relevant geometric properties are as provided in Alvarez-Vasquez (2009).

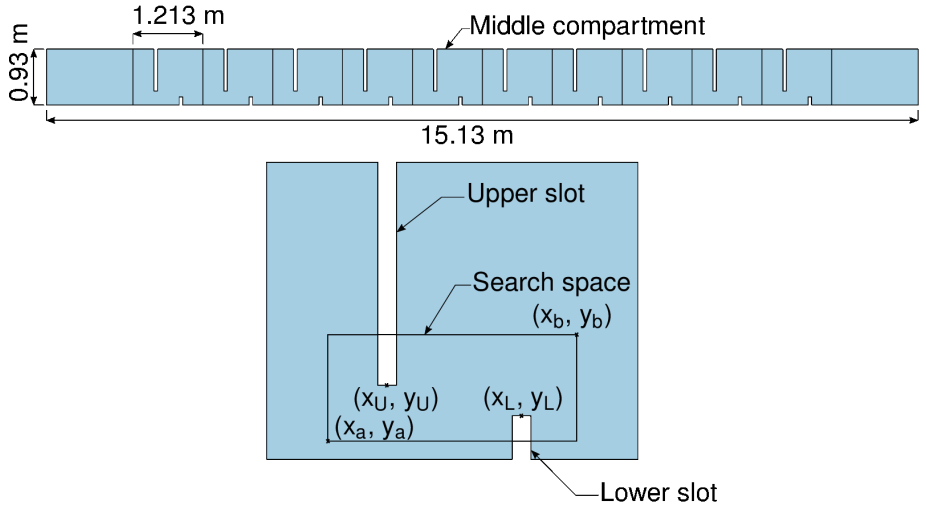


Figure 1: Geometry definition of the fish flume test case

3.2 Definition of the optimization problem: validation test case

The test case used in this paper focuses on validating the shape optimization tool chain developed. In this example the test case is defined as one that specifies a target (or desired) shape, and asks the optimizer to search the available solution space until the desired target is found. Retrieving the desired target is achieved by carefully defining the problem's objective function. The optimization process is then started with some initial state, with an end goal to recover the specified (or desired) shape. Such a procedure is implemented in the work by Goeury et al.(2017) when testing an automatic calibration procedure developed using the theory of data assimilation.

The shape of the desired state is specified with the target shape $(x_U, y_U, x_L, y_L)_t$, where the subscript t denotes the target shape. The geometry specified in the target shape is then used to construct a TELEMAC-2D model, and obtain a steady state solution. The tool chain developed is used to extract from the steady state solution of the target shape the x- and y- components of the flow velocity (u_t, v_t) in the middle compartment, and use it to construct a vector that stacks the values of velocity as follows:

$$y_{obs} = [u_t v_t] \quad (1)$$

If the middle compartment has n nodes, the number of elements of the y_{OBS} vector will thus be 2n. The observation vector is therefore defined as the state of the system (measured in terms of two velocity components) that the optimizer should recover after some number of iterations.

In this work, the optimizer will select and try new shapes during its course of execution. Each iteration of the optimizer requires a numerical solution of the flow field for a shape where different position of the slots are evaluated. When the optimizer completes a particular iteration (meaning computes a flow field for a particular shape), the tool chain extracts from the solution a vector y_{sim}, defined as:

$$y_{sim} = [u \quad v \quad] \tag{2}$$

where u and v are x- and y- components of the simulated velocity extracted from a particular shape from the middle compartment of the flume.

The optimization problem is then defined as follows:

$$\min g(x) = \frac{1}{n}\sum_{i=1}^{n}(y_{sim} - y_{obs})^2 \tag{3}$$

subject to:

$$(y_L - y_U) = \Delta y \tag{4}$$

$$(x_L - x_U) = \Delta x \tag{5}$$

where $x = (x_U, y_U, x_L, y_L)$ are the problem variables (position of the slots) and i = 1,2,3,...,n is the index of the nodes in the middle (i.e., fifth) compartment of the flume. Note that the optimization problem is bounded by a rectangular box with coordinates (x_a, y_a) on the lower left, and (x_b, y_b) on the upper right (see Figure 1).

The constraints of the problem are specified as the gap between the upper and lower groins. Without specifying the gap between the groins it would be possible for the optimizers to select the position of the groins where upper and lower slots touch or intersect, a physically meaningless configuration. The constrained optimization problem is thus defined such that the goal of the optimizer is to select valid position of both the upper and lower slots such that the objective function is minimized.

The particulars of the problem simulated here are (in units in meters, unless otherwise specified): Search space $x_a = 6.653$, $x_b = 7.260$, $y_a = 0.050$, and $y_b = 0.243$; width of the slots w = 0.063; channel slope $S_0 = 5\%$; bottom friction f = 57.36 m1/2/s; initial water depth $h_0 = 0.5$; discharge in the flume $Q_0 = 0.065$ m3/s; constraint values $\Delta x = 0.1$, and $\Delta y = 0.05$. The initial and target shapes are defined in Table 1.

Table 1: Validation test case: initial and target shapes

Shape	x_U [m]	y_U [m]	x_L [m]	y_L [m]	g(x) [-]
initial	6.700	0.230	7.230	0.150	0.137
target	6.927	0.147	7.168	0.054	0.0

The objective function defined above seeks to find a solution that is minimized. In case the optimizer can find a combination of (x_U, y_U, x_L, y_L) values that produce a value of the objective function g(x) = 0, it will exactly recover the desired (or target) shape. The optimizer that can do so in least number of iterations, will be the most favoured in practical applications.

If it can be demonstrated that the optimizers applied in this study can reach the target state (i.e., obtain a value of the objective function as close to zero as possible), the source code developed for the shape optimization tool chain can be validated as correct.

4 Results

Results of the fish flume shape optimization simulations are presented here, in tabular and graphical form. Note that even though the simulations in this work are completed for the entire flume (all ten compartments), the results are presented only for the fifth (i.e., middle) compartment. To carry out the validation of the tool chain, the initial shape of the flume is specified, as is the target shape (see Table 1). In the validation test case, the observation vector (y_{OBS}) is obtained by extracting the x- and y- velocity components of the target shape, simulated using TELEMAC-2D numerical model. Table 2 presents the summary of the optimization results for each optimizer used. Figure 2 shows the generated shape and its associated velocity field plot for the optima produced by select two optimizers, along with its convergence plots.

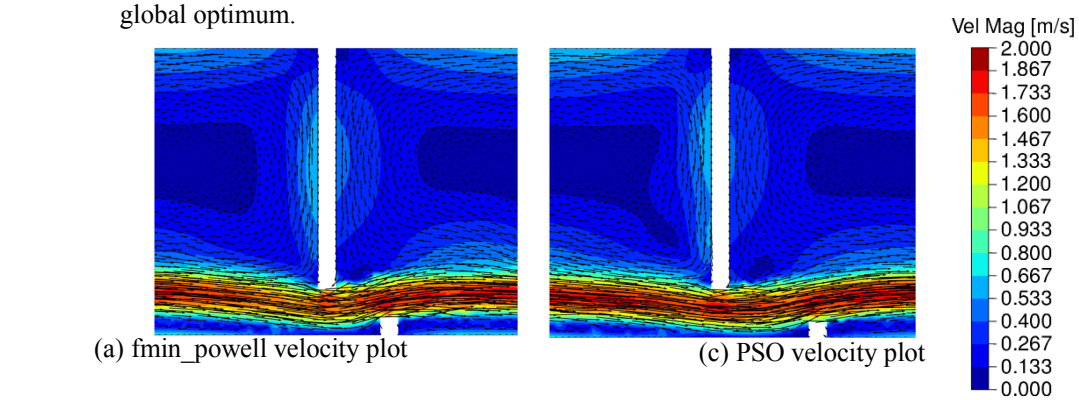
Table 2: Validation test case: summary of optimization results

Optimizer	x_U [m]	y_U [m]	x_L [m]	y_L [m]	g(x) [-]	Fun calls [count]
fmin	6.699	0.232	7.259	0.148	0.127	48
fmin_powell	6.935	0.152	7.146	0.062	0.003	170
buckshot	6.663	0.177	7.247	0.062	0.046	187
lattice	6.930	0.151	7.101	0.062	0.005	540
diffev	6.902	0.179	7.260	0.065	0.007	500
SLSQP	6.687	0.181	7.222	0.062	0.046	45
PSO	6.924	0.160	7.251	0.055	0.003	376
LHS	6.914	0.152	7.193	0.052	0.002	19,160

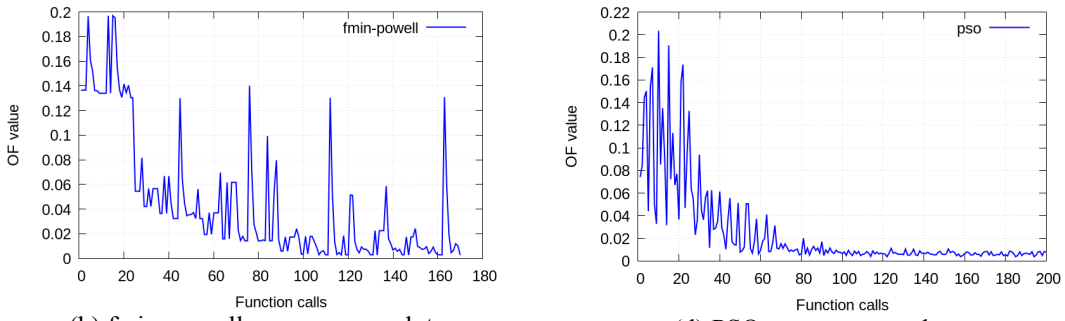
By inspecting the tabular and graphical results of the validation case a number of observations can be made. These are summarized below:

1. The fmin optimizer was not able to produce a satisfactory result, and was not able to recover the target shape.
2. Among all optimizers tested, the fmin_powell optimizer was able to generate an optimal solution (and thus recover the target shape) with the a good value of the objective function (i.e., one of the ones closest to zero).
3. The buckshot optimizer was able to produce a low value of the objective function, but it seemed to have converged to another local minimum (and did not find the exact target specified). However, the minimum found by the buckshot optimizer has a surprisingly similar velocity field as the target shape.
4. The solution of the lattice optimizer came close to reaching the target shape (the position of the lower slot in the shape generated by the optimizer is closer to the upper slot than in the target solution).

5. The diffev optimizer was able to produce a satisfactory solution after 500 function evaluations, with a fairly accurate optimal solution despite the fact that the position of the lower slot is not exactly at the location specified. It is possible that if the optimizer was allowed to proceed beyond 500 function evaluations, a better solution (i.e., closer to the target) could have been reached.
6. The results from the SLSQP optimizer, being the only gradient based optimizer, converged to another local minimum (similar to the buckshot optimizer), but with much less computational effort.
7. The PSO optimizer was able to find the optimum solution, with the least amount of computational effort. The optimum found is among the best obtained in the study.
8. The LHS method, also known as the brute force method, was able to recover the target shape, but with an extremely large number of function calls. The utility of the LHS method in this example is simply to demonstrate that other optimizers were able to find a global optimum.



(a) fmin_powell velocity plot (c) PSO velocity plot



(b) fmin_powell convergence plot (d) PSO convergence plot

The particulars of the current problem are such that the objective function used, when evaluated for distinct (but closely similar) shapes, produces similar numeric values. In other words, the objective function used is not sensitive to small fluctuation in the position of the slots. Having a different objective function (and one that is more sensitive to the position of the slots) is anticipated to be able to produce a result where the target shape would be recovered exactly by all of the optimizers tested.

5 Conclusions

This report presents a summary of the tool kit developed to study shape optimization problems in hydraulic engineering by coupling a numerical free surface flow solver with various numerical optimizers able to deal with non-linear non-convex optimization problems. The general methodology of shape optimization is presented, as is the summary of the tools developed. An example case study (optimum design of a fish passage) is provided for the purpose of testing the tools kit developed. The tool kit couples a numerical flow solver TELEMAC-2D to a number of numerical optimizers able to solve heavily constrained non-convex and non-linear problems. Seven such numerical optimizers were tested on a test case that required finding a solution of a four variable constrained optimization problem. Based on the results obtained, many of the optimizers were able to generate the optimum solution (and verified through a brute force calculation).

References

G. Elchahal, R. Younes, and P. Lafon (2013), Optimization of coastal structures: Application on detached breakwaters in ports, Ocean Engineering, 63, 35–43.

S. Erpicum, P. Archambeau, B. Dewals, M. Pirotton, C. Zhang and H. Tang (2009), Automatic geometrical optimization by way of numerical flow models. Advances in water resources and hydraulic engineering, Proceedings of 16th IAHR-APD Congress and 3rd Symposium of IAHR-ISHS, C. Zhang and H. Tang, eds., Springer-Verlag , 1663–1668.

L. Alvarez-Vasquez, A. Martinez, M. Vasquez-Mendez and M. Villar (2009), Numerical resolution of a shape optimization problem in hydraulic engineering, European Conference on Computational Fluid Dynamics, P. Wesseling and J. Periaux, eds., TU Delft, 1–13.

J. Nedler and R. Mead (1965), A simplex method for function minimization, Computer Journal, 7, 308–313.

P. Prodanovic (2017), PPUTILS: A practical toolkit for terrain, free surface flow and wave modeling, https://github.com/pprodano/pputils.

Scipy (2017), A Python-based ecosystem of open-source software for mathematics, science, and engineering. https://www.scipy.org.

Mystic (2017) ,Highly-constrained non-convex optimization and uncertainty quantification. https://github.com/uqfoundation/mystic.

Pyswarm (2017), Particle swarm optimization (PSO) with constraint support, https://github.com/tisimst/pyswarm.

M. McKerns, L. Strand, T. Sullivan, A. Fang, and M. Aivazis (2011), Building a framework for predictive science, Proceedings of the 10th Python in Science Conference, Austin, Texas.

C. Goeury, A. Poncot, J. Argaud, F. Zaoui, R. Ata and Y. Audouin (2017), Optimal calibration of TELEMAC-2D model based on a data assimilation algorithm, XXIVth TELEMAC-MASCARET User Conference, Graz University of Technology, Graz, Austria, 73–80.